AGI-Elo: How Far Are We From Mastering A Task?

Shuo Sun[1,3] **Yimin Zhao**[1] **Christina Dao Wen Lee**[1] **Jiawei Sun**[1] **Chengran Yuan**[1]
Zefan Huang[1,3] **Dongen Li**[1,3] **Justin KW Yeoh**[1] **Alok Prakash**[3]
Thomas W. Malone[2,3] **Marcelo H. Ang Jr.**[1,3]

[1]National University of Singapore [2]Massachusetts Institute of Technology
[3]Singapore MIT Alliance for Research and Technology
{shuo.sun,yimin.zhao,christinaldw,sunjiawei,chengran.yuan,
huangzefan,li.dongen}@u.nus.edu alok.prakash@smart.mit.edu
malone@mit.edu {justinyeoh,mpeangh}@nus.edu.sg

Abstract

As the field progresses toward Artificial General Intelligence (AGI), there is a pressing need for more comprehensive and insightful evaluation frameworks that go beyond aggregate performance metrics. This paper introduces a unified rating system that jointly models the difficulty of individual test cases and the competency of AI models (or humans) across vision, language, and action domains. Unlike existing metrics that focus solely on models, our approach allows for fine-grained, difficulty-aware evaluations through competitive interactions between models and tasks, capturing both the long-tail distribution of real-world challenges and the competency gap between current models and full task mastery. We validate the generalizability and robustness of our system through extensive experiments on multiple established datasets and models across distinct AGI domains. The resulting rating distributions offer novel perspectives and interpretable insights into task difficulty, model progression, and the outstanding challenges that remain on the path to achieving full AGI task mastery. We have made our code and results publicly available at `https://ss47816.github.io/AGI-Elo/`.

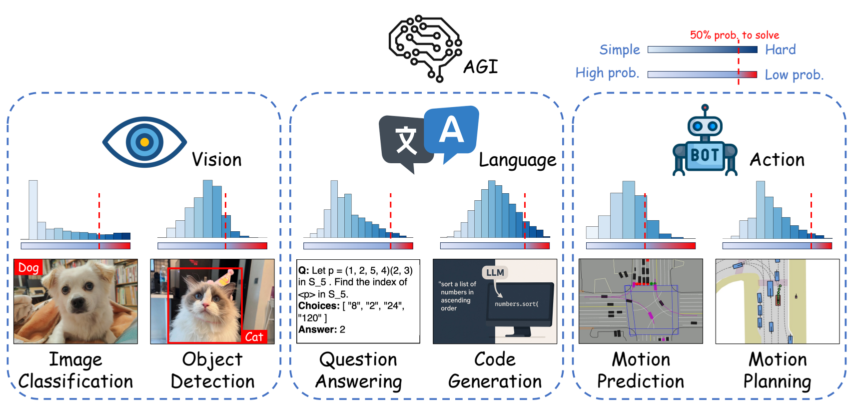


Figure 1: In this paper, we address long-standing questions regarding the current capabilities of AGI and humans on challenging tasks by proposing a standardized framework to quantitatively assess task difficulty, evaluate AGI competency, and identify gaps to task mastery.

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Track on Datasets and Benchmarks.

1 Introduction

As Artificial General Intelligence (AGI) begins to replace traditional Artificial Intelligence (AI) in our everyday lives, there is a growing need to systematically evaluate state-of-the-art (SOTA) AI models across a diverse range of tasks. These tasks span three fundamental domains: vision, language, and action. A crucial aspect of this evaluation is understanding not only the performance of AI models but also considering the inherent difficulty of the tasks they attempt to solve, and identifying the competency gap between the current models and the remaining unsolved difficult cases. As illustrated in Figure 1, this paper aims to quantitatively address three key questions simultaneously:

- What is the difficulty of each test case within a task or a dataset?
- What is the competency of an AI model (or a human) on a given task?
- How far are the current SOTA models from fully mastering a task?

1.1 Existing gaps in AI evaluation

Despite the extensive research on AI benchmarking, several fundamental gaps remain unaddressed:

Quantifying task and test case difficulties: Defining and measuring the difficulty of an entire task (e.g., a dataset) or an individual test case (e.g., a single image, question, or driving scenario) remains a fundamental challenge. While a range of heuristic proxies have been explored, such as curriculum learning signals [4], input characteristics [73, 24], training loss [26, 3, 70], model confidence [31], prediction variance [7], and information-theoretic measures [83]—these methods often rely on task-specific assumptions or model-dependent signals. A unified, systematic framework for quantifying difficulty consistent across tasks and interpretable from both AI and human perspectives is still lacking.

Difficulty-aware & predictive metric for AI: Most public benchmarks and datasets [48, 41, 13, 18, 6, 25, 17, 12] rely on task-specific metrics such as accuracy, mean Average Precision (mAP), and success rate to evaluate model performance. However, these metrics typically capture only aggregated performance across the dataset, providing relative comparisons between models rather than predictive indicators of how well an AI model (or a human) would perform on individual test cases of varying difficulty. This averaging effect obscures the underlying distribution of task difficulty and limits our understanding of a model's capacity to adapt to diverse and complex scenarios.

Progress over the long-tail in real-world tasks: Many real-world tasks exhibit a long-tail distribution, where certain test cases are significantly more challenging than others [86]. Identifying these difficult cases remains non-trivial, and existing benchmarks do not provide a systematic way to measure the length of the "tail", which is how much further AI models must progress before confidently claiming task mastery at a well-defined confidence interval (e.g., 50%, 90%, or 99%).

1.2 Our contributions

To address these gaps identified, we propose a rating system that jointly models task difficulty and model competency in a unified, probabilistic manner. Our key contributions are as follows:

1. **A task-agnostic rating system for AGI evaluations**: We introduce a rating system that simultaneously models test case difficulties and model competencies using a probabilistic approach. The rating of each test case or model is modeled as a normal distribution, which is constantly updated by a series of competitive matches between models and test cases.
2. **Unified measurement of test case difficulty and model competency**: Our framework provides a principled way to quantitatively estimate the difficulty of individual test cases and the comparative competency of intelligent agents (models or humans) simultaneously.
3. **Extensive experiments across domains**: Extensive experiments were conducted across the 3 AGI domains: vision, language, and action. To this end, we considered 6 well-established datasets using 7-20 models/humans that demonstrated effectiveness.
4. **Comprehensive evaluation and predictive insights**: By establishing a singular rating system for each of the AGI tasks, we analyze the rating distribution of test cases and the model ratings to identify the task difficulty distributions and long-tail characteristics. With this, we can conclude the competency gap from current models to fully mastering a task.

By establishing a robust and predictive rating system, our work provides a new perspective on AI evaluation, paving the way for a more comprehensive understanding of AI capabilities and limitations as we move toward AGI.

2 Rating systems explained

2.1 Conventional rating systems

Rating systems are commonly used to estimate the relative skill or performance of players based on outcomes of pairwise (or multiplayer) matches. After each match, the ranking system awards rating points to the winning side and deducts rating points from the losing side in a *zero-sum* fashion based on the match result.

Elo [15] is the foundational rating system originally developed for chess. It updates the ratings of both players based on the match score, assuming a logistic model of win probability. Given two players A and B with ratings R_A and R_B, the expected score of A can be computed as

$$\mathbb{E}[S_A] = \frac{1}{1 + 10^{(R_B - R_A)/400}}. \tag{1}$$

The rating update formula is given by:

$$R_A \leftarrow R_A + K(S_A - \mathbb{E}[S_A]), \tag{2}$$

where K is a sensitivity parameter and $S_A \in [0, 1]$ is the match score of A.

Glicko [19] extends the Elo system by modeling a player's rating as a Gaussian belief distribution characterized by a mean μ and a Rating Deviation (RD) σ, which quantifies the uncertainty in rating. Ratings with higher RD values are updated with a higher magnitude as compared to players with low RD, whose ratings will be more stable.

2.2 Properties & utilities

Probabilistic prediction: A key utility of rating systems is their predictive power. Given two players' ratings, the system can estimate the probability of each outcome based on Equation 1.

Translation-invariant: Rating systems are translation-invariant: shifting all ratings by a constant value does not affect the expected outcome. Only the relative difference in ratings between the two players determines the result, as the absolute scale is arbitrary and does not influence ranking behavior.

Transitivity: A desirable property of rating systems is transitivity: if player A consistently beats B, and B consistently beats C, then we expect A to have a higher rating than C. Transitivity enables the construction of a consistent global ranking across many players without requiring exhaustive pairwise evaluation.

Efficient placement: Only a small number of matches is required to determine the rating of a new player in the system. Efficient placement of new players with minimal evaluations is critical in large-scale settings.

3 AGI-Elo rating system design

The proposed *AGI-Elo* rating system consists of three main steps, including the conversion from benchmark results to match results, the update of models' and test cases' ratings based on match results, and the prediction of model competencies, as illustrated by the three arrows in Figure 2.

3.1 Test cases vs. agents

Conventional rating systems are primarily designed for *homogeneous* agents that can freely compete against one another in direct, one-on-one matches. In chess, humans and computers are assumed to be in the same category and compete directly, sharing comparable characteristics that make such matches meaningful.

Figure 2: Illustration of the proposed *AGI-Elo* rating system.

However, our proposed rating system diverges significantly as it is designed for matches between *heterogeneous* agents, in a similar fashion to Item Response Theory (IRT) [54], which models the probability that an agent (human or model) with a certain ability level correctly solves a test case as:

$$P(\text{correct} \mid \alpha, \beta, R_t, R_a) = \frac{1}{1 + \beta^{-\alpha(R_t - R_a)}} \tag{3}$$

where R_t and R_a present the difficulty of the test case and the ability of the agent, and $\alpha = 1/400$, $\beta = 10$ are assigned to follow existing conventions used in chess rating systems.

Specifically, *AGI-Elo* defines two distinct player types: **test cases** and **agents** (i.e., models or humans), and players can only engage in inter-category matches. A test case can be matched against an agent, but never *directly compete* with another test case; similarly, agents cannot compete with each other.

To enable the joint estimation of test case and agent ratings, *AGI-Elo* leverages the transitivity property of rating systems, under the assumption that the transitivity property remains valid in our *heterogeneous* agent setting (an assumption later supported by our experimental results in subsection 4.3). By observing the outcomes of inter-category matches, our rating system simultaneously infers ratings for both test cases and agents. Consequently, players within the same category are evaluated indirectly, with their relative ratings inferred through shared interactions with players from the opposing category.

Furthermore, our system explicitly incorporates the ratings of the intermediary category during the evaluation process. In particular, the rating of a test case plays a critical role in adjusting model ratings. For example, if a model fails on an easy (i.e., low-rated) test case, it is penalized more heavily than if it fails on a difficult (i.e., high-rated) one. By accounting for the inherent difficulty of each test case, the system avoids treating all errors equally, thereby preventing serious overestimation or underestimation of model competency in the presence of exceptionally easy or hard examples.

A key advantage of this rating system design is that model ratings are anchored to the empirical difficulty distribution of test cases. Moreover, the performance of any model on any test case can be quantitatively predicted.

3.2 Conversion to match results

For any given task, let $M \in \mathbb{R}$ denote a task-specific performance metric (e.g., accuracy, mAP), and let $f : \mathbb{R} \to [0,1]$ be a scoring function that maps M to a normalized match score $s \in [0,1]$. We define:

$$S = f(M) \tag{4}$$

The primary objective of the function f is to transform arbitrary task-specific metrics into a unified, continuous match score space, facilitating consistent comparison across matches. Once ratings are established in this normalized space, the inverse function $f^{-1} : [0,1] \to \mathbb{R}$ can be used to project predicted match scores back into the original metric space, yielding an interpretable predicted performance:

$$\hat{M} = f^{-1}(S) \tag{5}$$

To support generalization across diverse tasks and datasets, the scoring function f can be tailored to the specific characteristics and scale of the underlying metric M. This design ensures that our approach remains broadly applicable with minimal task-specific adjustments.

3.3 Rating update

To determine the appropriate rating adjustment after each match, we model the rating R of each player (whether a test case or a model) as a normal distribution $R \sim \mathcal{N}(\mu, \sigma^2)$ with a mean μ representing its rating score and a standard deviation σ representing the uncertainty in our estimate, following the Glicko system [19]. Initially, all models and test cases are assigned the same starting ratings. After each rated match, the μ and σ of both players are updated based on the match outcome. For each opponent j, the impact factor $g(\sigma_j)$, which adjusts the weight of the match outcome based on the opponent's uncertainty, is defined as:

$$g(\sigma_j) = \frac{1}{\sqrt{1 + \frac{3q^2\sigma_j^2}{\pi^2}}} \tag{6}$$

where $q = \frac{\ln(10)}{400} \approx 0.0057565$. The expected outcome of player i against opponent j is:

$$E_{ij} = \frac{1}{1 + 10^{-g(\sigma_j)(\mu_i - \mu_j)/400}} \tag{7}$$

After a rated match where player i competes against multiple opponents j, the new rating is updated as:

$$\mu_i \leftarrow \mu_i + \frac{q}{\frac{1}{\sigma_i^2} + \sum_j g(\sigma_j)^2 E_{ij}(1 - E_{ij})} \sum_j g(\sigma_j)(S_{ij} - E_{ij}) \tag{8}$$

where $S_{ij} \in [0,1]$ represents the actual match score. The updated rating deviation is given by:

$$\sigma_i \leftarrow \left(\frac{1}{\sigma_i^2} + \sum_j g(\sigma_j)^2 E_{ij}(1 - E_{ij}) \right)^{-1/2} \tag{9}$$

After a sufficient number of matches, ideally when all models have competed against all test cases, the ratings of both models and test cases should converge to stable values that reflect their respective competency and difficulty levels.

3.4 Prediction

With the ratings of both models and test cases determined, we can leverage the properties of the rating system to make the following predictions:

Agent performance: The expected performance $\mathbb{E}[M_a]$ of an agent a in the original metric space on a test case t can be estimated as:

$$\mathbb{E}[M_a] = f^{-1}(\mathbb{E}[S_a]) = f^{-1}\left(\frac{1}{1+10^{(R_t-R_a)/400}}\right), \tag{10}$$

where $\mathbb{E}[S_a]$ denotes the expected match outcome of agent a, and R_a, R_t represent the mean rating values of the agent and the test case, respectively.

Long-tailed test cases beyond an agent's competency: The set of test cases on which agent a is expected to achieve a performance below a threshold M_θ (in the original metric space) is defined as:

$$\mathcal{T}^{\text{hard}}_{a,M_\theta} = \left\{ t \in \mathcal{T} \;\middle|\; f^{-1}\left(\frac{1}{1+10^{(R_t-R_a)/400}}\right) < M_\theta \right\}, \tag{11}$$

where $\mathcal{T}$ denotes the complete set of test cases in the dataset.

Oracle's task mastery levels: In AI and machine learning, an *oracle* typically refers to a model that achieves ideal performance or provides ground-truth answers for a given task. In the context of this paper, the concept of an "oracle" serves solely as a theoretical reference point, illustrating where future models with higher skill levels might be positioned relative to current models. Assuming the dataset is a faithful miniature reflection of the real-world distribution of test cases, the oracle's performance on the most difficult test case in the dataset serves as a proxy for its worst-case performance in the real world. In this paper, we further quantify an oracle using either a performance threshold S_θ in the match score space or a corresponding threshold M_θ in the original metric space. The hypothetical oracles with different confidence levels and their ratings can be estimated based on the distribution of the test cases in the post-experiment analysis. For example, a hypothetical *oracle @ M_θ mastery* is defined as a model capable of achieving at least M_θ performance, or equivalently, at least $S_\theta \times 100\%$ confidence in solving, all test cases in the task. The rating required for such an oracle can be estimated as:

$$R_{\textit{oracle@}S_\theta} \geq R_{t,\text{max}} - 400 \cdot \log_{10}\left(\frac{1-S_\theta}{S_\theta}\right), \tag{12}$$

where $R_{t,\text{max}} = \max\{R_t \mid t \in \mathcal{T}\}$ denotes the rating of the hardest test case in the dataset.

Agent's competency gap to full task mastery: The competency gap for an agent a to reach this oracle-level performance is defined as:

$$\text{Competency Gap} = R_{\textit{oracle@}S_\theta} - R_a, \tag{13}$$

which quantifies how much the agent's rating must improve in order to achieve the desired level of task mastery.

4 Experiments

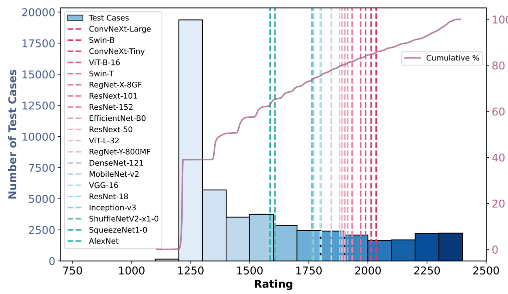
4.1 Experimental setup

We selected six representative tasks spanning three core AGI domains—vision, language, and action: image classification, object detection, question answering, code generation, motion prediction, and motion planning. For each task, we chose the most widely adopted dataset: ImageNet [13], COCO [48], MMLU [29], LiveCodeBench [37], Waymo [17], and NAVSIM [12], respectively.

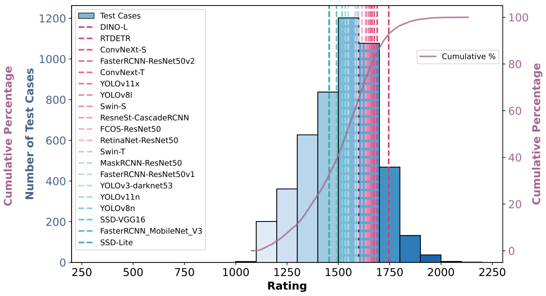
The specific agents evaluated, as well as the evaluation metrics and scoring functions used for each dataset, are detailed in Appendix B. Notably, the motion planning task includes a *human expert* as one of the evaluated agents, alongside AI models. All players (both agents and test cases) are initialized with a rating of $R \sim \mathcal{N}(1500, 350^2)$. During the rating update step, the order of matches is fully randomized to ensure smooth and unbiased convergence of ratings.

4.2 Rating distributions

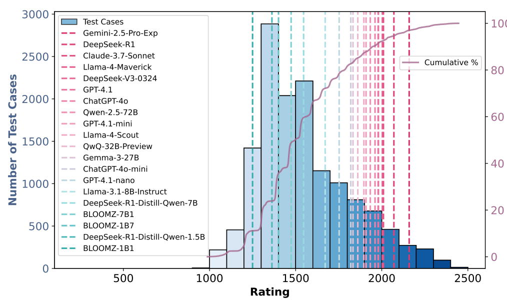
In Figure 3, the rating distributions of both test cases and agents are visualized across all six datasets. To provide a **qualitative evaluation** of test case difficulty, we randomly sample test cases from each rating level for every dataset/task and present them in Appendix A for visual comparison.



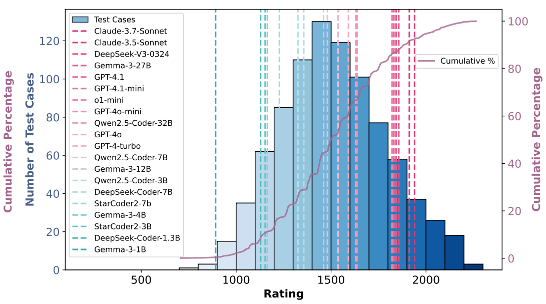
(a) Image classification: ImageNet [13]



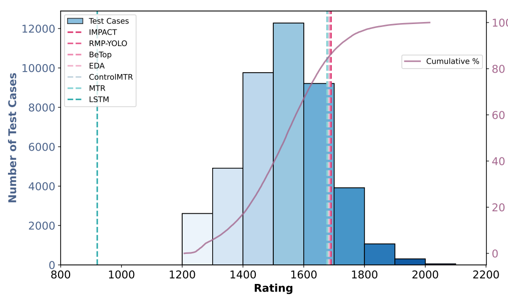
(b) Object detection: COCO [48]



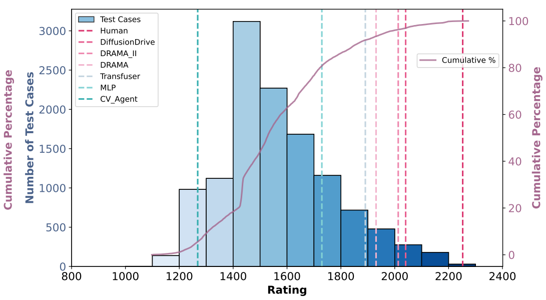
(c) Question answering: MMLU [29]



(d) Code generation: LiveCodeBench [37]



(e) Motion prediction: Waymo [17]



(f) Motion planning: NAVSIM [12]

Figure 3: Visualization of the estimated test case rating distribution and agent ratings on six distinct datasets. The percentile curve represents the cumulative percentage of test cases up to each rating level. For each agent, the portion of the test cases and the percentile curve that lies to the right represents the fraction of the dataset that remains difficult (below 50% confidence).

From Figure 3, we observe distinct test case difficulty distributions across different datasets by examining the histograms and the percentile curves over the rating spectrum. Datasets such as ImageNet [13], MMLU [29], and NAVSIM [12] exhibit long-tail distributions, indicated by a small fraction of highly challenging test cases. In contrast, LiveCodeBench [37] and Waymo [17] present more symmetrical distributions from the agents' perspectives, indicating a more balanced spread of difficulty levels. Meanwhile, COCO [48] shows a short-tail distribution, suggesting that its most difficult test cases are relatively moderate in comparison.

By observing the improvements in model performance, we can trace the progress made on each task over the years. For example, on ImageNet [13], ConvNeXt-Large [53] (2022) obtained a rating of 2035, successfully surpassing approximately 85% of test images (rated < 2035) with at least 50% confidence, and about 67% of images (rated $< 1635 = 2035 - 400$) with at least 91% confidence. Compared to AlexNet [42] (2012), which beats 64% of the dataset with a rating of 1586, the progress over 10 years is about 449 rating points, and newly mastering 18% of the dataset.

Table 1: Competency gaps estimated on each dataset (excluding human)

Domain	Task	Dataset	Metric	$R_{t,\max}$	$R_{a,\max}$	$\mathbb{E}[M_{a,t}]\uparrow$	Competency Gap to Oracles ↓		
							@50%	@90%	@99%
Vision	Classification	ImageNet [13]	Acc@1	2389.7	2035.0	0.115	354.7	736.4	1152.9
	Detection	COCO [48]	AP@[.50:.90]	2132.7	1745.5	0.097	387.2	768.9	1185.4
Language	QA	MMLU [29]	Accuracy	2446.1	2159.2	0.161	286.9	668.6	1085.1
	Coding	LiveCodeBench [37]	PassAll	2263.3	1939.7	0.134	323.6	705.3	1121.8
Action	Prediction	Waymo [17]	mAP	2014.3	1689.8	0.134	324.5	706.2	1122.8
	Planning	NAVSIM [12]	PDM Score	2273.0	2040.5	**0.208**	**232.5**	**614.2**	**1030.8**

In Table 1, we report the highest-rated agents and test cases for each dataset, along with the predicted expected performance of each agent on the most difficult test case and the corresponding competency gaps to oracles at various confidence thresholds.

The results show that, excluding the human agent, the highest-rated AI models across the six datasets generally exhibit competency gaps of approximately 233–387 rating points from achieving mastery on the most difficult test cases at the 50% confidence level, and approximately 1031–1185 rating points from oracles @ 99% confidence level. In contrast, the human expert on the NAVSIM [12] dataset achieves near-oracle-level competency under the PDM score metric, with a gap of only 20.7 rating points from the oracle @ 50% confidence. This suggests that the human agent is approaching the performance of an ideal oracle on this task. These findings highlight that, in the presence of challenging test cases, current AI models remain significantly below oracle-level performance and face substantial competency gaps that must be bridged before achieving true task mastery.

4.3 Reliability of the rating system

As the proposed method is uniquely designed for rating *heterogeneous* players, it is essential to evaluate the reliability of the resultant ratings to ensure meaningful interpretations and to validate the assumptions underlying the design of the rating system. We assess rating reliability from two key perspectives: **consistency** with existing evaluation metrics and **predictive accuracy**.

Consistency: Spearman's rank correlation is used to measure the consistency between our estimated rating rankings and the original task-specific performance metrics. For each test case t, we record the average agent performance $\bar{M}_t$ on that test case, and for each agent a, we compute the average agent performance $\bar{M}_a$ across all test cases. The Spearman's rank correlation coefficient ρ_t between the rankings of $\{R_t\}$ and $\{\bar{M}_t\}$, and ρ_a between the rankings of $\{R_a\}$ and $\{\bar{M}_a\}$, are used as indicators.

Predictive accuracy: For each agent, its average performance $\bar{M}_{a,B} = \frac{1}{|B|}\sum_{t\in B} M_{a,t}$ on all test cases within the same rating bin B is computed and compared against the theoretical expectations $\mathbb{E}[M_{a,B}]$ derived from the rating system. The mean absolute error (MAE) and mean squared error (MSE) are used to quantify the deviation between the empirical performance $\bar{M}_{a,B}$ and the theoretical expectation $\mathbb{E}[M_{a,B}]$.

Table 2: Consistency & predictive accuracy across various datasets

Dataset	Split	N_t	N_a	N_{match}	Consistency		Predictive Accuracy	
					$\rho_t\downarrow$	$\rho_a\uparrow$	MAE ↓	MSE↓
ImageNet [13]	val	50,000	20	1,000,000	-0.9685	0.9985	0.0476	0.0039
COCO [48]	val	4,952	20	99,040	-0.9999	1.0000	0.0167	0.0005
MMLU [29]	test	13,957	20	279,140	-0.9962	1.0000	0.0662	0.0076
LiveCodeBench [37]	test	880	20	17,600	-0.9968	0.9985	0.0446	0.0038
Waymo [17]	val	44,097	7	308,679	-0.9981	1.0000	0.0354	0.0023
NAVSIM [12]	test	12,147	7	85,050	-0.9963	1.0000	0.0546	0.0088

As shown in Table 2, our method achieves consistently low MAE and MSE across all datasets, highlighting its accuracy in ratings and predictive performance. Experimental results also demonstrate that our method achieves consistently high correlation, indicating a strong association between the derived ratings and the traditional aggregate metrics. Despite the strong overall correlation, our approach uniquely uncovers subtle rank-reversal cases, where models with similar traditional scores

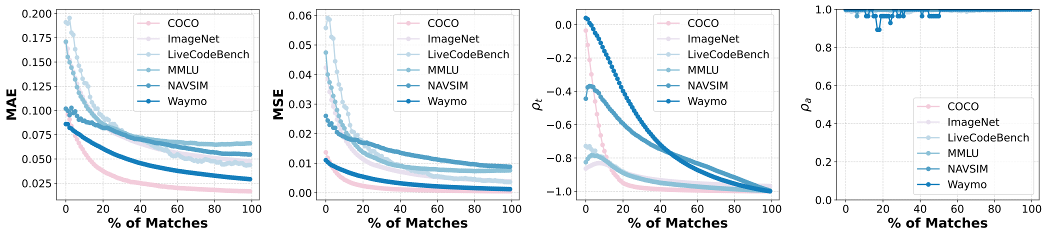


Figure 4: Evaluation of the reliability as a function of the percentage of completed matches.

receive different relative rankings under our method. The Spearman's rank correlation coefficient ρ_a on both ImageNet [13] and LiveCodeBench [37] is 0.9985, instead of perfectly 1, indicating the existence of such "rank-reversal" cases in the model rankings. More specifically, on the ImageNet [13] dataset, the "rank-reversal" case happened between ViT-B-16 (Acc. 0.81066, rating 1969.5) and Swin-T (Acc. 0.81088, rating 1969.0); while on the LiveCodeBench [37] dataset, the "rank-reversal" case happened between GPT-4.1-mini (Acc. 0.77019, rating 1832.1) and o1-mini (Acc. 0.77361, rating 1820.5).

In Figure 4, we plot the evolution of all four evaluation metrics as a function of the percentage of matches used by the rating system. As more match data is incorporated, both MAE and MSE consistently decrease, indicating the convergence and stability of the system. Similarly, the correlation grows stronger with additional match information, demonstrating the effectiveness of our method in accurately rating both test cases and agents. These trends provide empirical support for the transitivity assumption introduced earlier.

5 Related works

Estimating per-instance difficulty Evaluating instance difficulties in datasets is an important yet understudied field [85, 77]. Some methods rely on hand-crafted features like word overlap [4], input length [73, 24], or similarity scores [60] as proxies for difficulty, which are oversimplistic. Many techniques adopt loss-based metrics [26, 3, 70] or prediction confidence [31, 7, 83]. Approaches like [82, 77, 16] leverage model training dynamics, which can offer deeper insights, but are often influenced by the stochastic nature of training. However, these methods often yield model-specific difficulty estimates that are difficult to compare across models due to varying loss designs, and they are typically inapplicable to non-learning agents like classical algorithms or human agents. In contrast, our system directly utilizes performance metrics as difficulty indicators, making it broadly compatible and capable of capturing insights from a wide range of agents. This universality ensures that the estimated difficulties are meaningful and comparable across different agent types.

Benchmarking AI capabilities Inspired by competitive games, several works have adopted rating systems to evaluate AI model performance across tasks or in head-to-head comparisons. For example, rating systems have been used to assess AlphaStar agents in StarCraft II competitions [84] and in reinforcement learning tournaments [30]. The Chatbot Arena framework [9] applies a modified Elo system to conduct pairwise comparisons of large language models (LLMs), based on crowd-sourced human preference judgments. However, these evaluation approaches typically focus solely on modeling agent capabilities, without accounting for the implicit difficulty of individual test cases. As a result, the estimated model ratings may fail to reflect true performance under varying levels of difficulty and can be unreliable [5]. Furthermore, such model-vs-model competition setups are not easily generalizable to a wide range of AI tasks beyond dialogue or games.

Psychometric benchmarks [92, 44] have also been applied to the AI domain to assess question difficulty and model ability. In particular, Item Response Theory (IRT) has been adapted to characterize the relative competency of models across tasks and datasets, enabling fine-grained performance profiling [58]. However, prior works have primarily focused on basic machine learning tasks with simple classifiers, without extending to a broad range of complex tasks and state-of-the-art (SOTA) models. By integrating rating systems with IRT-inspired evaluation, our framework offers a unified and interpretable approach to jointly estimate test case difficulty and model competency. This enables more reliable predictions for models on tasks, while preserving generalizability.

6 Conclusion and limitations

In this paper, we propose *AGI-Elo*, a unified framework for jointly estimating task difficulties and agent competencies through a quantifiable, general-purpose rating system tailored for AGI tasks. Experimental results across six diverse tasks spanning vision, language, and action domains demonstrate the broad applicability and high predictive accuracy of our approach. The resulting rating distributions enable in-depth analysis of dataset difficulty characteristics, precise identification of long-tailed challenging test cases, and quantification of competency gaps between current AI agents and idealized oracles at various levels. To support further research, we release the computed test case and agent ratings, and we hope that our findings will stimulate broader interest in this important yet underexplored area.

While our results offer a novel perspective, they are not exhaustive. Due to limited computational resources, our current experimental scale is constrained, and the selected datasets and models may not fully represent state-of-the-art performance. Nevertheless, we believe the proposed methodology is sound, and we envision future studies expanding upon it with more comprehensive evaluations across the full spectrum of AGI capabilities.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Anthropic. Claude 3.7 sonnet and claude code. `https://www.anthropic.com/news/claude-3-7-sonnet`, February 2025. Accessed: 2025-05-16.

[3] Eric Arazo, Daniel Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Unsupervised label noise modeling and loss correction. In *International Conference on Machine Learning*, pages 312–321, 2019.

[4] Yoshua Bengio, Jerome Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[5] Meriem Boubdir, Edward Kim, Beyza Ermis, Sara Hooker, and Marzieh Fadaee. Elo uncovered: Robustness and best practices in language model evaluation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[6] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020.

[7] Hwanjun Songkuk Chang and et al. Active bias: Training more accurate neural networks by emphasizing high variance samples. In *Advances in Neural Information Processing Systems*, pages 1002–1012, 2017.

[8] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, et al. Mmdetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019.

[9] Wei-Lin Chiang, Lianmin Zheng, Ying Sheng, Anastasios Nikolas Angelopoulos, Tianle Li, Dacheng Li, Banghua Zhu, Hao Zhang, Michael Jordan, Joseph E Gonzalez, et al. Chatbot arena: An open platform for evaluating llms by human preference. In *Forty-first International Conference on Machine Learning*, 2024.

[10] Kashyap Chitta, Aditya Prakash, Bernhard Jaeger, Zehao Yu, Katrin Renz, and Andreas Geiger. Transfuser: Imitation with transformer-based sensor fusion for autonomous driving. *Pattern Analysis and Machine Intelligence (PAMI)*, 2023.

[11] OpenScene Contributors. Openscene: The largest up-to-date 3d occupancy prediction benchmark in autonomous driving, 2023.

[12] Daniel Dauner, Marcel Hallgarten, Tianyu Li, Xinshuo Weng, Zhiyu Huang, Zetong Yang, Hongyang Li, Igor Gilitschenski, Boris Ivanovic, Marco Pavone, Andreas Geiger, and Kashyap Chitta. Navsim: Data-driven non-reactive autonomous vehicle simulation and benchmarking. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[15] Arpad E Elo. The proposed uscf rating system, its development, theory, and applications. *Chess life*, 22(8):242–247, 1967.

[16] Kawin Ethayarajh, Yejin Choi, and Swabha Swayamdipta. Understanding dataset difficulty with v-usable information. In *International Conference on Machine Learning*, pages 5988–6008. PMLR, 2022.

[17] Scott Ettinger, Shuyang Cheng, Benjamin Caine, Chenxi Liu, Hang Zhao, Sabeek Pradhan, Yuning Chai, Ben Sapp, Charles R Qi, Yin Zhou, et al. Large scale interactive motion forecasting for autonomous driving: The waymo open motion dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9710–9719, 2021.

[18] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[19] Mark E. Glickman. Parameter estimation in large dynamic paired comparison experiments. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 48(3):377–394, 1999.

[20] Google DeepMind. Gemini 2.5: Our most intelligent ai model. `https://blog.google/technology/google-deepmind/gemini-model-thinking-updates-march-2025/`, March 2025. Accessed: 2025-05-16.

[21] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

[22] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[23] Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Yu Wu, YK Li, et al. Deepseek-coder: When the large language model meets programming–the rise of code intelligence. *arXiv preprint arXiv:2401.14196*, 2024.

[24] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 2, pages 107–112, 2018.

[25] K. Tan et al. H. Caesar, J. Kabzan. Nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles. In *CVPR ADP3 workshop*, 2021.

[26] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems*, pages 8527–8537, 2018.

[27] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. *arXiv preprint arXiv:1703.06870*, 2017.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[29] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

[30] Ralf Herbrich, Tom Minka, and Thore Graepel. Trueskill™: a bayesian skill rating system. *Advances in neural information processing systems*, 19, 2006.

[31] Dirk Hovy, Barbara Plank, and Anders Sogaard. Learning whodunnit: Classification of event participants in news articles. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 540–545, 2013.

[32] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[33] Binyuan Hui, Jian Yang, Zeyu Cui, Jiaxi Yang, Dayiheng Liu, Lei Zhang, Tianyu Liu, Jiajun Zhang, Bowen Yu, Keming Lu, et al. Qwen2. 5-coder technical report. *arXiv preprint arXiv:2409.12186*, 2024.

[34] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[35] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016.

[36] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[37] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

[38] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolov8, 2023.

[39] Glenn Jocher and Jing Qiu. Ultralytics yolo11, 2024.

[40] Glenn Jocher, Alex Stoken, Jirka Borovec, Liu Changyu, Adam Hogan, Laurentiu Diaconu, Jake Poznanski, Lijun Yu, Prashant Rai, Russ Ferriday, et al. ultralytics/yolov5: v3. 0. *Zenodo*, 2020.

[41] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[42] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 25, pages 1097–1105, 2012.

[43] Quentin Lhoest, Albert Villanova Del Moral, Yacine Jernite, Abhishek Thakur, Patrick Von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, et al. Datasets: A community library for natural language processing. *arXiv preprint arXiv:2109.02846*, 2021.

[44] Yuan Li, Yue Huang, Hongyi Wang, Xiangliang Zhang, James Zou, and Lichao Sun. Quantifying ai psychology: A psychometrics benchmark for large language models. *arXiv preprint arXiv:2406.17675*, 2024.

[45] Bencheng Liao, Shaoyu Chen, Haoran Yin, Bo Jiang, Cheng Wang, Sixu Yan, Xinbang Zhang, Xiangyu Li, Ying Zhang, Qian Zhang, and Xinggang Wang. Diffusiondrive: Truncated diffusion model for end-to-end autonomous driving. *arXiv preprint arXiv:2411.15139*, 2024.

[46] Longzhong Lin, Xuewu Lin, Tianwei Lin, Lichao Huang, Rong Xiong, and Yue Wang. Eda: Evolving and distinct anchors for multimodal motion prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 3432–3440, 2024.

[47] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *arXiv preprint arXiv:1708.02002*, 2017.

[48] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer vision–ECCV 2014: 13th European conference, zurich, Switzerland, September 6-12, 2014, proceedings, part v 13*, pages 740–755. Springer, 2014.

[49] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[50] Haochen Liu, Li Chen, Yu Qiao, Chen Lv, and Hongyang Li. Reasoning multi-agent behavioral topology for interactive autonomous driving. In *NeurIPS*, 2024.

[51] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 21–37. Springer, 2016.

[52] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022, 2021.

[53] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022.

[54] Frederic M Lord and Melvin R Novick. *Statistical theories of mental test scores*. IAP, 2008.

[55] Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, et al. Starcoder 2 and the stack v2: The next generation. *arXiv preprint arXiv:2402.19173*, 2024.

[56] Wenyu Lv, Shangliang Xu, Yian Zhao, Guanzhong Wang, Jinman Wei, Cheng Cui, Yuning Du, Qingqing Dang, and Yi Liu. Detrs beat yolos on real-time object detection (2023). *arXiv preprint arXiv:2304.08069*, 2023.

[57] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, and Jian Sun. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In *Proceedings of the European conference on computer vision (ECCV)*, pages 116–131, 2018.

[58] Fernando Martínez-Plumed, Ricardo BC Prudêncio, Adolfo Martínez-Usó, and José Hernández-Orallo. Item response theory in ai: Analysing machine learning classifiers at the instance level. *Artificial intelligence*, 271:18–42, 2019.

[59] Meta AI. The llama 4 herd: The beginning of a new era of natively multimodal ai innovation. `https://ai.meta.com/blog/llama-4-multimodal-intelligence/`, April 2025. Accessed: 2025-05-16.

[60] Swaroop Mishra, Anjana Arunkumar, Chris Bryan, and Chitta Baral. Hardness of samples need to be quantified for a reliable evaluation system: Exploring potential opportunities with a new task. *arXiv preprint arXiv:2210.07631*, 2022.

[61] Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, et al. Crosslingual generalization through multitask finetuning. *arXiv preprint arXiv:2211.01786*, 2022.

[62] NanoGPT. Nanogpt api. `https://nano-gpt.com/api`, 2025. Accessed: 2025-05-15.

[63] OpenAI. Openai api. https://platform.openai.com, 2025. Accessed: 2025-05-15.

[64] A Paszke. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

[65] Qwen Team. Qwq: Reflect deeply on the boundaries of the unknown. https://qwenlm.github.io/blog/qwq-32b-preview/, November 2024. Accessed: 2025-05-16.

[66] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10428–10436, 2020.

[67] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[68] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *arXiv preprint arXiv:1506.01497*, 2015.

[69] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4510–4520, 2018.

[70] Yao Shen and Sujay Sanghavi. Learning with bad training data via iterative trimmed loss minimization. In *International Conference on Machine Learning*, pages 5739–5748, 2019.

[71] Shaoshuai Shi, Li Jiang, Dengxin Dai, and Bernt Schiele. Motion transformer with global intention localization and local movement refinement. *Advances in Neural Information Processing Systems*, 2022.

[72] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[73] Valentin I Spitkovsky, Hiyan Alshawi, and Dan Jurafsky. Baby steps: How "less is more" in unsupervised dependency parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 751–759, 2010.

[74] Jiawei Sun, Jiahui Li, Tingchen Liu, Chengran Yuan, Shuo Sun, Zefan Huang, Anthony Wong, Keng Peng Tee, and Marcelo H Ang Jr. Rmp-yolo: A robust motion predictor for partially observable scenarios even if you only look once. *arXiv preprint arXiv:2409.11696*, 2024.

[75] Jiawei Sun, Chengran Yuan, Shuo Sun, Shanze Wang, Yuhang Han, Shuailei Ma, Zefan Huang, Anthony Wong, Keng Peng Tee, and Marcelo H. Ang. Controlmtr: Control-guided motion transformer with scene-compliant intention points for feasible motion prediction. In *2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1507–1514, 2024.

[76] Jiawei Sun, Xibin Yue, Jiahui Li, Tianle Shen, Chengran Yuan, Shuo Sun, Sheng Guo, Quanyun Zhou, and Marcelo H Ang Jr. Impact: Behavioral intention-aware multimodal trajectory prediction with adaptive context trimming. *arXiv preprint arXiv:2504.09103*, 2025.

[77] Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. Dataset cartography: Mapping and diagnosing datasets with training dynamics. *arXiv preprint arXiv:2009.10795*, 2020.

[78] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[79] Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6105–6114, 2019.

[80] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

[81] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9627–9636, 2019.

[82] Mariya Toneva, Alessandro Sordoni, Yulia Tsvetkov, Tommi Jaakkola, and Ellie Pavlick. An empirical study of example forgetting during deep neural network learning. In *International Conference on Learning Representations*, 2019.

[83] Neeraj Varshney, Swaroop Mishra, and Chitta Baral. Ildae: Instance-level difficulty analysis of evaluation data. *arXiv preprint arXiv:2203.03073*, 2022.

[84] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[85] Kumar Vodrahalli, Ganesh Ramakrishnan, and Balaraman Ravindran. Are all training examples created equal? an empirical study. In *arXiv preprint arXiv:1803.07156*, 2018.

[86] Haohui Wang, Weijie Guan, Jianpeng Chen, Zi Wang, and Dawei Zhou. Towards heterogeneous long-tailed learning: Benchmarking, metrics, and toolbox. In *Advances in Neural Information Processing Systems 37 (NeurIPS 2024), Datasets and Benchmarks Track*, 2024.

[87] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[88] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, et al. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*, 2024.

[89] Chengran Yuan, Zhanqi Zhang, Jiawei Sun, Shuo Sun, Zefan Huang, Christina Dao Wen Lee, Dongen Li, Yuhang Han, Anthony Wong, Keng Peng Tee, et al. Drama: An efficient end-to-end motion planner for autonomous driving with mamba. *arXiv preprint arXiv:2408.03601*, 2024.

[90] Hang Zhang, Chongruo Wu, Zhongyue Zhang, Yi Zhu, Zhi Zhang, Haibin Lin, Yue Sun, Tong He, Jonas Muller, R. Manmatha, Mu Li, and Alexander Smola. Resnest: Split-attention networks. *arXiv preprint arXiv:2004.08955*, 2020.

[91] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. *arXiv preprint arXiv:2203.03605*, 2022.

[92] Yan Zhuang, Qi Liu, Yuting Ning, Weizhe Huang, Zachary A Pardos, Patrick C Kyllonen, Jiyun Zu, Qingyang Mao, Rui Lv, Zhenya Huang, et al. From static benchmarks to adaptive testing: Psychometrics in ai evaluation. *arXiv preprint arXiv:2306.10512*, 2023.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction describes the proposed rating system (AGI-Elo), the experiments done to validate and the generalizability and reliability of AGI-Elo (Section 4, Appendix A, B).

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The limitations are discussed in the Section 6, the conclusion section.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: The paper does not include theoretical results

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The code will be open-sourced on Github and the datasets used in the experiment will be available through HuggingFace. The procedures to run the experiment will also be described on Github.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: The code will be provided on Github.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Experiment settings and details are released together with the code on Github.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper does not provide error bars due to the expensive nature of the resources needed to run the large scale experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The compute resources used are specified in Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The NeurIPS Code of Ethics has been reviewed. Since the research described in this paper does not involve humans and is not believed to have potentially harmful social impacts, it is assumed to conform to the NeurIPS code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [No]

    Justification: The paper describes a technical method to rate different AGI models in completing tasks in common domains. There are no specific discussions about societal impacts other than the beliefs that the rating system will be helpful, implying positive social impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper describes a new method to benchmark AGI capabilities, the risk for misuse is believed to be low.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The papers describing the dataset used are cited. The specific datasets source, versions, and license will be described on the public dataset repository on HuggingFace.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The code used will be published on Github under CC BY-NC-SA 4.0. Documentation will be provided in the repository.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The research described in the paper does not involve crowdsourcing or human test subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The research described in this paper does not involve crowdsourcing or human subjects, therefore no risk disclosure or IRB approval is needed.

    Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are being evaluated by our rating system. Although it is a core component of our research, it is not modified in any way in this paper.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.

A Supplementary results for experiments

A.1 Qualitative evaluation

We have made our qualitative examples available on:

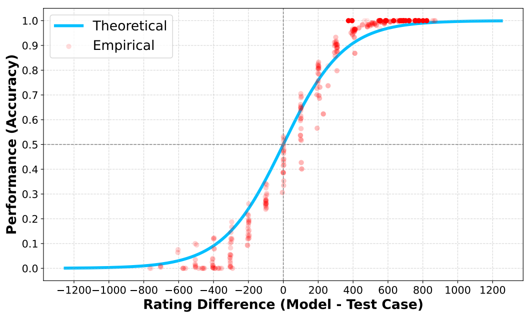
HuggingFace:

`https://huggingface.co/collections/ztony0712/agi-elo-6825d88e9587700e9dd41b12`

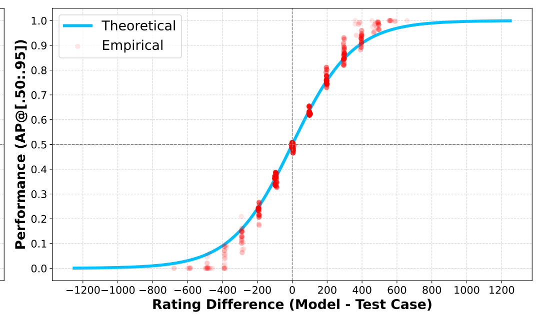
Project page:

`https://ss47816.github.io/AGI-Elo/`

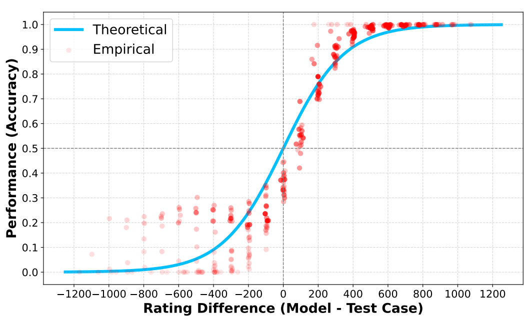
A.2 Performance prediction vs. reality: predictive accuracy on various datasets



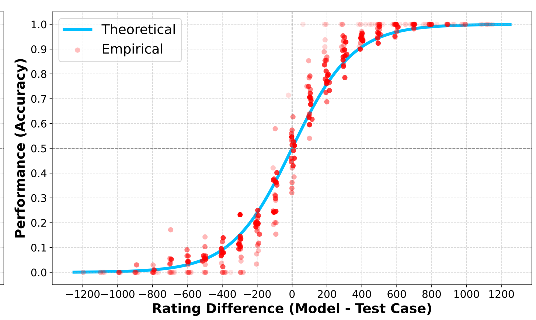
(a) Image classification: ImageNet [13]



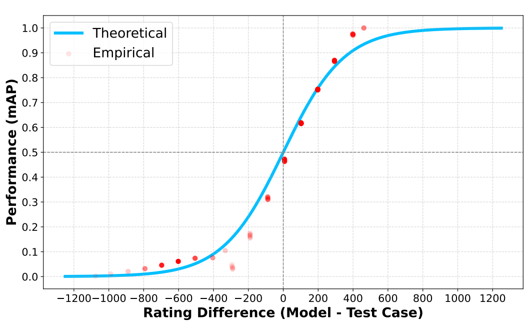
(b) Object detection: COCO [48]



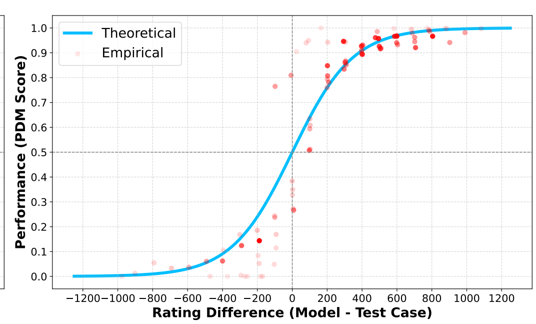
(c) Question answering: MMLU [29]



(d) Code generation: LiveCodeBench [37]



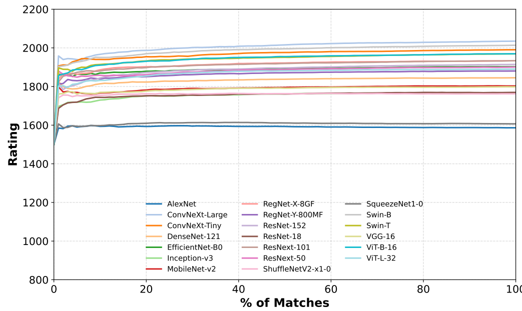
(e) Motion prediction: Waymo [17]



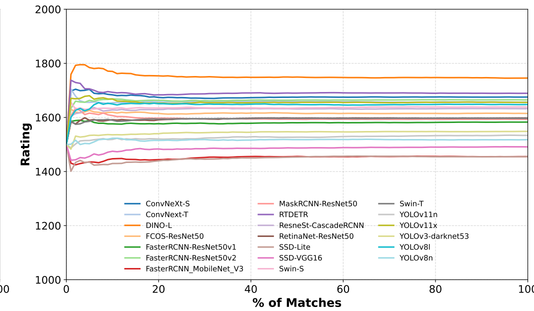
(f) Motion planning: NAVSIM [12]

Figure 5: Visualization of the predicted (theoretical) agent performances based on the differences between agents and test cases vs. the empirical performance obtained on each dataset.

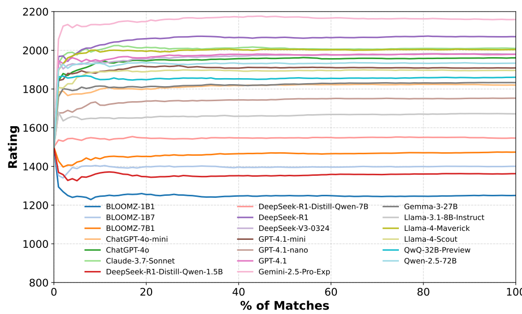
A.3 Influence of match percentage on model rating stability



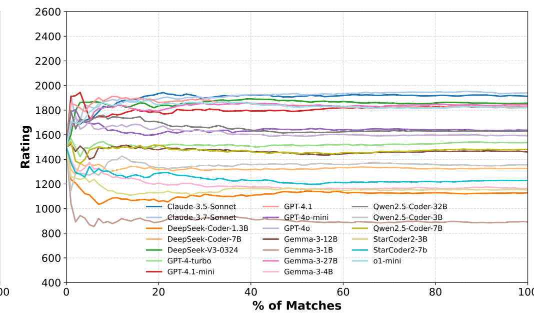
(a) Image classification: ImageNet [13]



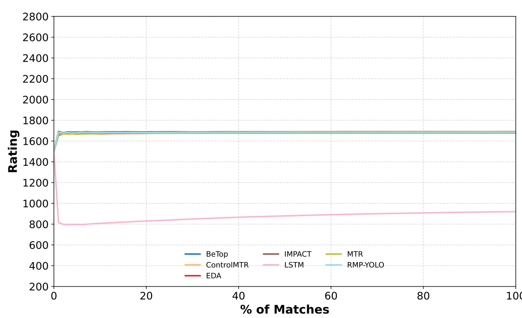
(b) Object detection: COCO [48]



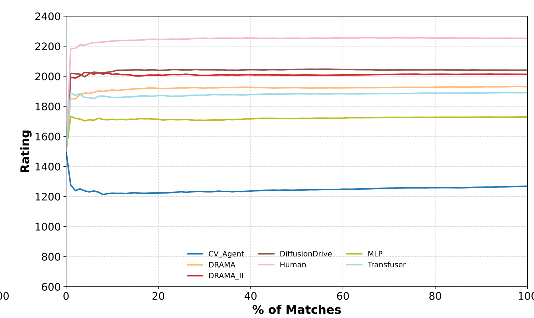
(c) Question answering: MMLU [29]



(d) Code generation: LiveCodeBench [37]



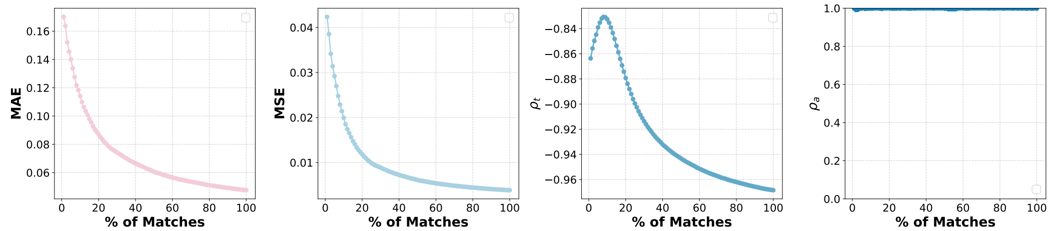
(e) Motion prediction: Waymo [17]



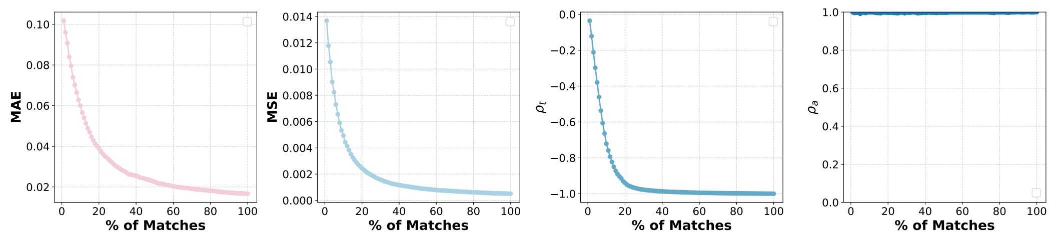
(f) Motion planning: NAVSIM [12]

Figure 6: Model ratings over the percentage of matches on respective datasets.

A.4 Effect of percentage of matches on rating system accuracy and consistency

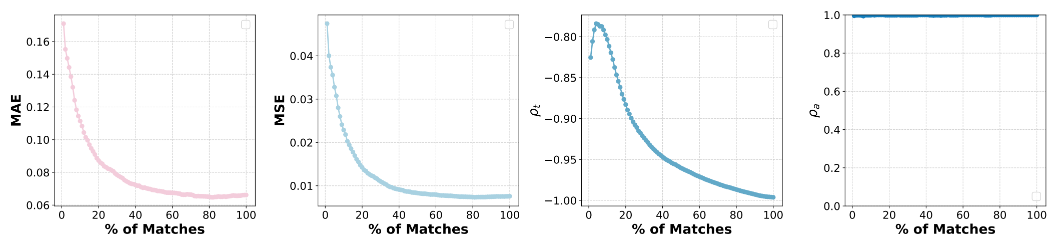


(a) Image classification: ImageNet [13]

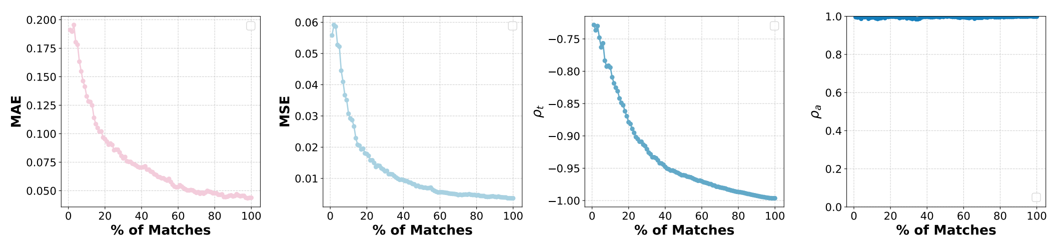


(b) Object detection: COCO [48]

Figure 7: System prediction errors and Spearman's correlations over the percentage of matches on respective datasets (Vision).

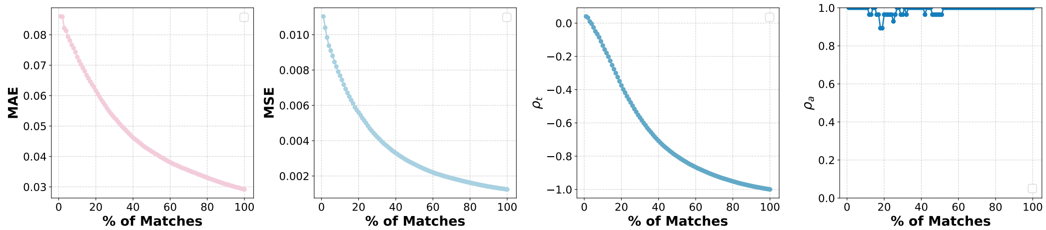


(a) Question answering: MMLU [29]

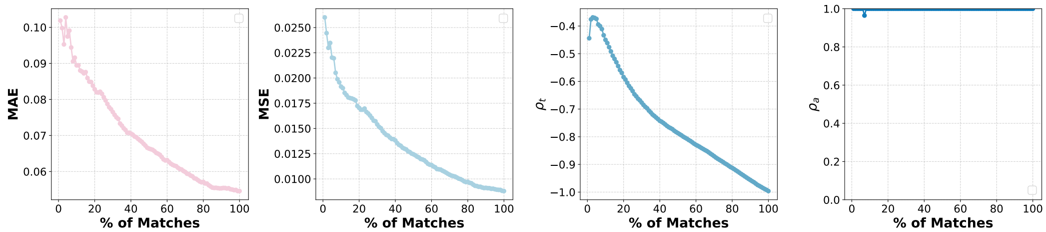


(b) Code generation: LiveCodeBench [37]

Figure 8: System prediction errors and Spearman's correlations over the percentage of matches on respective datasets (Language).



(a) Motion prediction: Waymo [17]



(b) Motion planning: NAVSIM [12]

Figure 9: System prediction errors and Spearman's correlations over the percentage of matches on respective datasets (Action).

B Detailed experimental setup

B.1 Vision - Image Classification

B.1.1 Dataset

For the computer vision task, we selected the ImageNet [13] dataset, which is one of the most widely used and challenging public benchmarks for image classification. The dataset consists of over 14 million labeled images spanning 1,000 object categories. Experiments were conducted on the validation set, which contains 50,000 distinct images, ensuring a diverse and comprehensive evaluation of model performance.

B.1.2 Metric

On the ImageNet [13] dataset, the standard Acc@1 metric is used:

$$\text{Acc@1} = \frac{1}{N}\sum_{i=1}^{N} 1\left(\hat{y}_i = y_i\right) \tag{14}$$

B.1.3 Scoring function

The scoring function f used on the ImageNet [13] dataset is defined as:

$$S := \text{Acc@1} \tag{15}$$

B.1.4 Models

On the image classification task, we selected 20 representative image classification models and summarized their key characteristics and release years in Table 3. All pretrained models were obtained from the `torchvision.models` module in PyTorch [64] and evaluated on a local desktop equipped with an Intel i9-12900K CPU, 32 GB of RAM, and an NVIDIA RTX 3090 Ti GPU.

Table 3: Image classification models

#	Model	Year	Source
1	ConvNeXt-Large [53]	2022	Pytorch
2	Swin-B [52]	2021	Pytorch
3	ConvNeXt-Tiny [53]	2022	Pytorch
4	ViT-B-16 [14]	2020	Pytorch
5	SwinT [52]	2021	Pytorch
6	RegNet-X-8GF [66]	2020	Pytorch
7	ResNext-101 [87]	2017	Pytorch
8	ResNet-152 [28]	2016	Pytorch
9	EfficientNet-B0 [79]	2019	Pytorch
10	ResNext-50 [87]	2017	Pytorch
11	ViT-L-32 [14]	2020	Pytorch
12	RegNet-Y-800MF [66]	2020	Pytorch
13	DenseNet-121 [32]	2017	Pytorch
14	MobileNet-v2 [69]	2018	Pytorch
15	VGG16 [72]	2014	Pytorch
16	ResNet-18 [28]	2016	Pytorch
17	Inception-v3 [78]	2016	Pytorch
18	ShuffleNetV2-x1-0 [57]	2018	Pytorch
19	SqueezeNet1-0 [35]	2016	Pytorch
20	AlexNet [42]	2012	Pytorch

B.2 Vision - Object Detection

B.2.1 Dataset

Object Detection is a task that started almost a decade ago. To this end, we use the established dataset and benchmark the COCO dataset [48], evaluating on the validation set, which consists of 5,000 images.

B.2.2 Metric

Based on the 2017 validation split (val2017) evaluation guidelines, the metric used, AP:0.5-0.95, was calculated by averaging AP over 80 object classes AND all 10 IoU thresholds from 0.5 to 0.95 with a step size of 0.05 as shown in Equation 16.

$$\text{AP}_{\text{COCO}} = \frac{1}{10} \sum_{k=0}^{9} \text{AP}_{\text{IoU}=0.50+0.05k} \tag{16}$$

B.2.3 Scoring function

The scoring function f used on the Waymo dataset is defined as:

$$S := \text{AP}_{\text{COCO}} \tag{17}$$

B.2.4 Models

Similar to the image classification task, we selected 20 object detection models that vary in performance and year of development. They constitute models that have developed over the years. The models include: models with a CNN vs a Transformer backbone, and vary in speed and performance.

All pretrained models were obtained from PyTorch [64], MMDetection [8], and Ultralytics [40], and evaluated on a local desktop equipped with an Intel i9-12900K CPU, 32 GB of RAM, and an NVIDIA RTX 3090 Ti GPU.

Table 4: Object detection models

#	Model	Year	Source
1	DINO-L [91]	2023	MMDetection
2	RT-DETR [56]	2023	Ultralytics
3	ConvNeXt-S [53]	2022	MMDetection
4	Faster R-CNN- ResNet50 -v2 [68]	2015	PyTorch
5	ConvNeXt-T [53]	2022	MMDetection
6	YOLOv11x [39]	2024	Ultralytics
7	YOLOv8l [38]	2023	Ultralytics
8	Swin-S [52]	2021	MMDetection
9	ResNeSt [90]	2021	MMDetection
10	FCOS [81]	2019	PyTorch
11	RetinaNet [47]	2017	PyTorch
12	Swin-T [52]	2021	MMDetection
13	MaskRCNN [27]	2017	MMDetection
14	Faster RCNN -ResNet50 - v1 [68]	2015	PyTorch
15	YOLOv3 [67]	2018	MMDetection
16	YOLOv11n [39]	2024	Ultralytics
17	YOLOv8n [38]	2023	Ultralytics
18	SSD-VGG16 [51]	2016	PyTorch
19	Faster R-CNN -MobileNetv3 [68]	2015	PyTorch
20	SSDLite [51]	2016	PyTorch

B.3 Language - Question Answering

B.3.1 Dataset

The MMLU (Massive Multitask Language Understanding) benchmark [29] is designed to evaluate models on a diverse set of challenging tasks that span 57 subjects, including mathematics, history, law, and computer science. To this end, we evaluate models on the official test split, which contains multiple-choice questions with four options each.

B.3.2 Metrics

Following the original evaluation protocol, we report the Acc@1 metric, defined as the proportion of questions for which the model selects the correct answer, as shown in Equation 18. This metric captures the model's ability to perform zero-shot reasoning across a wide range of knowledge-intensive tasks.

$$\mathrm{Acc@1} = \frac{1}{N}\sum_{i=1}^{N} 1\left(\hat{y}_i = y_i\right) \tag{18}$$

where $\hat{y}_i$ denotes the model's predicted answer for the i-th question, and $1(\hat{y}_i = y_i)$ is an indicator function that returns 1 if the prediction matches the ground truth y_i, and 0 otherwise.

B.3.3 Scoring function

The scoring function f used on the Waymo dataset is defined as:

$$S := \mathrm{Acc@1} \tag{19}$$

B.3.4 Models

For this task, we selected 20 LLMs that vary in performance and year of development. The three BLOOMZ [61] pretrained models were obtained from Huggingface [43], and evaluated on a local desktop equipped with an Intel i9-12900K CPU, 32 GB of RAM, and an NVIDIA RTX 3090 Ti GPU. The other models were evaluated using the OpenAI API [63] and the NanoGPT API [62] online.

Table 5: Question answering models

#	Model	Year	Source
1	Gemini-2.5-Pro-Exp [20]	2025	NanoGPT API
2	DeepSeek-R1 [22]	2025	NanoGPT API
3	Claude-3.7-Sonnet [2]	2025	NanoGPT API
4	Llama-4-Maverick [59]	2025	NanoGPT API
5	DeepSeek-V3-0324 [49]	2025	NanoGPT API
6	GPT-4.1 [1]	2025	OpenAI API
7	GPT-4o [34]	2024	OpenAI API
8	Qwen2.5-72B [88]	2024	NanoGPT API
9	GPT-4.1-mini [1]	2025	OpenAI API
10	Llama-4-Scout [59]	2025	NanoGPT API
11	QwQ-32B-Preview [65]	2024	NanoGPT API
12	Gemma-3-27B [80]	2025	NanoGPT API
13	GPT-4o-mini [34]	2024	OpenAI API
14	GPT-4.1-nano [1]	2025	OpenAI API
15	Llama-3.1-8B-Instruct [21]	2024	NanoGPT API
16	DeepSeek-R1-Distill-Qwen-7B [22]	2025	NanoGPT API
17	BLOOMZ-7B1 [61]	2023	hf (bigscience/bloomz-7b1)
18	BLOOMZ-1B7 [61]	2023	bigscience/bloomz-1b7
19	DeepSeek-R1-Distill-Qwen-1.5B [22]	2025	NanoGPT API
20	BLOOMZ-1B1 [61]	2023	bigscience/bloomz-1b1

B.4 Language - Code Generation

B.4.1 Dataset

LiveCodeBench is a recently proposed benchmark for evaluating the live code generation capabilities of large language models. To this end, we adopt the livecodebench/code_generation_lite dataset [37], which comprises executable, interactive coding problems designed to simulate real-world programming tasks. Evaluation is conducted on the 5th version of the official test split, which contains 880 problems spanning diverse domains such as algorithms and data structures.

B.4.2 Metric

Following the evaluation protocol outlined by the authors, each model is assessed based on Functional Correctness (FC), defined as the proportion Equation 20 of generated code completions that pass all test cases for a given problem.

$$\mathrm{FC} = \frac{1}{N}\sum_{i=1}^{N} 1\left(\texttt{PassAll}(\hat{c}_i)\right) \tag{20}$$

where $\texttt{PassAll}(\hat{c}_i)$ is an indicator function that returns 1 if the generated code $\hat{c}_i$ passes all functional test cases for the i-th problem, and 0 otherwise.

B.4.3 Scoring function

The scoring function f used on the Waymo dataset is defined as:

$$S := \mathrm{PassAll}(\hat{c}_i) \tag{21}$$

B.4.4 Models

For the code generation task, we selected 20 LLMs known for their strong performance in programming-related benchmarks. Several pretrained models were obtained from Huggingface [43], and evaluated on a local desktop equipped with an Intel i9-12900K CPU, 32 GB of RAM, and an NVIDIA RTX 3090 Ti GPU. The other models were evaluated using the OpenAI API [63] and the NanoGPT API [62] online.

Table 6: Code generation models

#	Model	Year	Source
1	Claude-3.7-Sonnet [2]	2025	NanoGPT API
2	Claude-3.5-Sonnet [2]	2024	NanoGPT API
3	DeepSeek-V3-0324 [49]	2025	NanoGPT API
4	Gemma-3-27B [80]	2025	NanoGPT API
5	GPT-4.1 [1]	2025	OpenAI API
6	GPT-4.1-mini [1]	2025	OpenAI API
7	o1-mini [36]	2024	OpenAI API
8	GPT-4o-mini [34]	2024	OpenAI API
9	Qwen2.5-Coder-32B [33]	2024	NanoGPT API
10	GPT-4o [34]	2024	OpenAI API
11	GPT-4-turbo [1]	2024	OpenAI API
12	Qwen2.5-Coder-7B [33]	2024	hf (Qwen/Qwen2.5-Coder-7B-Instruct)
13	Gemma-3-12B [80]	2025	hf (google/gemma-3-12b-it)
14	Qwen2.5-Coder-3B [33]	2024	hf (Qwen/Qwen2.5-Coder-3B-Instruct)
15	DeepSeek-Coder-7B [23]	2024	hf (deepseek-ai/deepseek-coder-7b-instruct)
16	StarCoder2-7B [55]	2024	hf (bigcode/starcoder2-7b)
17	Gemma-3-4B [80]	2025	hf (google/gemma-3-4b-it)
18	StarCoder2-3B [55]	2024	hf (bigcode/starcoder2-3b)
19	DeepSeek-Coder-1.3B [23]	2024	hf (deepseek-ai/deepseek-coder-1.3b-instruct)
20	Gemma-3-1B [80]	2025	hf (google/gemma-3-1b-it)

B.5 Action - motion prediction

B.5.1 Dataset

For the motion prediction task, we adopt the Waymo Open Motion Dataset (WOMD) [17], one of the most comprehensive and challenging public datasets for autonomous driving behavior prediction. WOMD is specifically designed to facilitate research on multi-agent trajectory forecasting in complex urban environments. The dataset contains a total of 486,995 training clips, 44,097 validation clips, and 44,920 testing clips. Each clip spans 8 seconds and is recorded at a sampling frequency of 10 Hz. Within each clip, 10 timesteps of historical agent states, 1 current timestep, and 80 future timesteps are provided, enabling both short-term and long-term trajectory forecasting. Evaluation is conducted on the validation split using the official Waymo evaluation API. For each selected target agent (as specified by Waymo), the model generates six candidate future trajectories along with their associated confidence scores.

B.5.2 Metric

In the WOMD, there are eight predefined trajectory buckets, including straight, straight-left, straight-right, left, right, left u-turn, right u-turn, and stationary [17]. For each bucket, a predicted trajectory is classified as a false positive if it is considered a miss as defined in MR; otherwise, it is classified as a true positive. Consistent with the mAP metrics used in object detection tasks, a maximum of one true positive is assigned to the one with the highest probability, while all others are assigned a false positive. True positives and false positives are then stored by their probabilities, and a Precision / Recall (P/R) curve can be plotted for each bucket. The Average Precision (AP) is represented by the area under the P/R curve, and the mAP metric can be computed by averaging the AP across all buckets as:

$$\text{mAP} = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{22}$$

B.5.3 Scoring function

The scoring function f used on Waymo [17] dataset is defined as:

$$S := \text{mAP} \tag{23}$$

B.5.4 Models

To ensure a fair and consistent evaluation, we reproduced all listed motion prediction models using a unified hardware setup consisting of eight NVIDIA RTX 3090 GPUs. For the publicly available models, we followed their official open-source implementations closely, adapting only minor components where necessary to ensure compatibility within our evaluation framework. As ControlMTR and IMPACT are not publicly available, we contacted the authors directly and received assistance in replicating their results.

Table 7: Motion prediction models

#	Model	Year	Source
1	Waymo LSTM Baseline [17]	2021	Proprietary
2	MTR [71]	2022	https://github.com/sshaoshuai/MTR
3	EDA [46]	2023	https://github.com/Longzhong-Lin/EDA
4	ControlMTR [75]	2023	Proprietary
5	RMP-YOLO [74]	2024	https://github.com/ggosjw/RMP-YOLO
6	BETOP [50]	2024	https://github.com/OpenDriveLab/BeTop
7	IMPACT [76]	2025	Proprietary

B.6 Action - motion planning

B.6.1 Dataset

To evaluate the motion planning performance, we adopt the NAVSIM benchmark [12], which utilizes the OpenScene dataset [11] - a refined derivative of the nuPlan [25]. This comprehensive benchmark features 120 hours of vehicle trajectories sampled at 2Hz, providing multimodal sensor observations including: (1) synchronized 8-view high-resolution RGB image (1920×1080 pixels) and (2) fused LiDAR point clouds aggregated from five sensors. The agent's input encompasses the current observation frame along with three temporally preceding frames, thereby providing 1.5 seconds of continuous temporal context. For quantitative evaluation of the closed-loop planning performance, we employ the Predictive Driver Model Score (PDMS) provided in the NAVSIM benchmark.

B.6.2 Metric

The PDMS in NAVSIM v1.1 is formulated as follows:

$$\text{PDMS} = \text{NC} \times \text{DAC} \times \frac{(5 \times \text{EP} + 5 \times \text{TTC} + 2 \times \text{C})}{12}, \tag{24}$$

where NC (no collision), DAC (driving area compliance), EP (ego progress), TTC (time-to-collision), and C (comfort) are sub-metrics as detailed in [12].

B.6.3 Scoring function

The scoring function f used on the NAVSIM dataset is defined as:

$$S := \text{PDMS} \tag{25}$$

B.6.4 Models

On the motion planning task, we reproduced all motion prediction models using the same hardware setup consisting of eight NVIDIA RTX 3090 GPUs. For the publicly available models, we followed their official open-source implementations closely to ensure a fair and consistent evaluation. As DRAMA II is not publicly available, we contacted the authors directly and received assistance in replicating their results.

Table 8: Motion planning models

#	Model	Year	Source
1	Human [12]	-	NAVSIM Ground Truth
2	DiffusionDrive [45]	2025	https://github.com/hustvl/DiffusionDrive
3	DRAMA II	2025	Proprietary
4	DRAMA [89]	2024	https://chengran-yuan.github.io/DRAMA/
5	Transfuser [10]	2024	https://github.com/autonomousvision/transfuser
6	MLP	2023	https://github.com/autonomousvision/navsim
7	CV Agent	2000	https://github.com/autonomousvision/navsim